Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of the Claims:

1. (Currently Amended) A method of creating data structures suitable for use by a virtual machine to execute computer instructions, the method comprising: converting a stream of **virtual machine** commands and data associated with the **virtual machine** commands into a pair of streams for use in the virtual machine, the pair of streams including a code stream **and a data stream, wherein the code stream** ~~that~~ includes [[the]] **virtual machine** commands, and **wherein** [[a]] **the** data stream ~~that~~ includes the data associated with the **virtual machine** commands in the code stream.

2. (Original) A method as recited in claim 1, wherein the code stream includes only commands and the data stream includes only the data associated with the commands in the code stream.

3. (Previously Presented) A method as recited in claim 1, wherein the stream that is to be converted is a Java™ compliant bytecode stream, the code stream is a Java™ bytecode code stream that includes Java™ commands, and the data stream is a Java™ bytecode data stream that includes the data associated with the Java™ commands in the Java™ bytecode code stream.

4. (Currently Amended) A method as recited in claim 1, wherein said converting of said stream comprises:

writing a representation of a first command associated with a first instruction into a code entry of the code stream;

determining whether the first command has data associated with it; and

writing a representation of the associated data or a reference to a representation of the data associated with the first command into a first data entry of the data stream when the command has ~~an~~ associated data.

5. (Previously Presented) A method as recited in claim 4, wherein the stream that is to be converted is a Java™ bytecode stream, the code stream is a Java™ bytecode code stream that includes Java™ commands, and the data stream is a Java™ bytecode data stream that includes the data associated with the Java™ commands in the Java™ bytecode code stream.

6. (Original) A method as recited in claim 4, wherein said method further comprises:

not providing a data entry in the data stream for the command when the command does not have data associated with it.

7. (Previously Presented) A method as recited in claim 6, wherein said method further comprises:

writing a representation of a second command associated with another instruction into a second code entry of the code stream;

determining whether the second command has data associated with it; and

writing a representation of the associated data or a reference to a representation of the data associated with the second command into a second data entry of the data stream when the command has associated data.

8. (Previously Presented) A method as recited in claim 7, wherein the stream that is to be converted is a Java™ bytecode stream, the code stream is a Java™ bytecode code stream that includes Java™ commands, and the data stream is a Java™ bytecode data stream that includes the data associated with the Java™ commands in the Java™ bytecode code stream.

9. (Previously Presented) A method as recited in claim 8,

wherein the command and data entries can each include a number of bytecodes in the Java™ bytecode stream,

wherein the number of bytecodes is an integer, and

wherein each byte code can be one or more bytes.

10. (Original) A method as recited in claim 9, wherein the first and second entries of the code stream are adjacent to each other.

11. (Previously Presented) In an object oriented programming environment, a computer readable medium including computer program code for generating computer executable commands and data associated with the computer executable commands suitable for use by a virtual machine and comprising:

computer program code for receiving a first stream of virtual machine instructions that include virtual machine commands and data associated with the commands;

computer program code for generating, from the first stream, a code stream having one or more virtual machine commands and

computer program code for generating, from the first stream, a data stream having data associated with the one or more virtual machine commands.

12. (Previously Presented) A computer readable medium as recited in claim 11, wherein the code stream does not include any data associated with the virtual machine commands and the data stream does not include any virtual machine commands.

13. (Previously Presented) A computer readable medium as recited in claim 11, wherein the code stream includes Java™ commands represented as bytecodes, and the data stream that includes the data associated with the Java™ commands represented as bytecodes.

14. (Previously Presented) A computer readable medium as recited in claim 13,

wherein the code stream and the data stream each comprise of a plurality of Java™ bytecodes representing a Java™ bytecode stream,

wherein each Java™ command and data associated with that command are represented respectively in one or more bytecodes of the code stream and the data stream, and

wherein each bytecode can be one or more bytes.

15. (Previously Presented) A computer readable medium as recited in claim 14, wherein the Java™ commands can be a load constant command, an invoke method command, a jump command, an instantiation command, or a get/put field command.

16. (Previously Presented) A method of executing computer instructions on a virtual machine, the method comprising:

fetching a command associated with a virtual machine computer instruction from a code stream;

determining whether the command has an associated parameter;

fetching from a data stream the associated parameter of the command when said determining determines that command has an associated parameter; and

executing the command with the associated parameters after the associated parameter of the commands have been fetched.

17. (Original) A method as recited in claim 16, wherein the method further comprises:

updating a pointer to the command stream; and

updating a pointer to the data stream.

18. (Previously Presented) A method as recited in claim 16, wherein the code stream includes Java™ compliant commands represented as bytecodes, and the code stream includes data associated with the Java™ commands represented as bytecodes.

19. (Previously Presented) A method as recited in claim 18,

wherein the code stream and the data stream each comprise of a plurality of Java™ byte codes representing a Java™ bytecode stream,

wherein each Java™ command and data associated with each Java™ command are represented respectively, in one or more bytecodes of the code stream and the data stream, and

wherein each bytecode can be one or more bytes.

20. (Previously Presented) A method as recited in claim 19, wherein the Java™ commands can be a load constant command, an invoke method command, a jump command, an instantiation command, or a get/put field command.

21. (Previously Presented) A method of creating data structures suitable for use by a virtual machine to execute Java™ instructions, the method comprising:

converting a Java™ compliant bytecode into a pair of Java™ bytecode streams suitable for use by the virtual machine, the pair of Java™ bytecode streams having a Java™ code stream that includes Java™ commands and a Java™ data stream that includes the data associated with the commands included in the code stream

22. (Previously Presented) A method as recited in claim 21, wherein said converting of said stream comprises:

writing a representation of a first command associated with a first instruction into a code entry of the code stream;

determining whether the first command has associated data; and

reading the associated data from a Constant Pool when the command has an associated data;

processing the associated data from the Constant Pool the associated data from the Constant Pool;

writing a representation of the associated data or a reference to a representation of the data associated into the Java™ code stream after said processing of the associated data;

wherein each Java™ command and data associated with that Java™ command are represented respectively in one or more bytecodes of the Java™ code stream and the Java™ data stream, and

wherein each bytecode can be one or more bytes.

23. (Previously Presented) A method as recited in claim 22, wherein said processing operates to determine a constant value associated with a Java™ Load Constant command.

24. (Previously Presented) A method as recited in claim 22, wherein said processing operates to determine a reference to a method invocation cell that includes information relating to a Java™ invoke method command.

25. (Previously Presented) A method as recited in claim 22, wherein said processing operates to determine the code stream offset and data stream offset associated with a Java™ Jump command.

26. (Previously Presented) A method as recited in claim 22, wherein said processing operates to process a Constant Pool associated with a Java™ instantiation command.

27. (Previously Presented) A method as recited in claim 22, wherein said processing operates to process a Constant Pool associated with a Java™ Get/Put field command.

28. (Previously Presented) A method as recited in claim 22, wherein said processing operates to process data associated with a Java™ load constant command, a Java™ invoke method command, a Java™ jump command, a Java™ instantiation command, or a Java™ get/put field command.

29. (Previously Presented) A method of executing a virtual machine instruction on a virtual machine, the method comprising:

reading a virtual machine instruction form a first stream; wherein the virtual machine instruction has a code portion and one or more data portions the

converting the virtual machine instruction into a pair of streams, the pair of streams consisting of a code stream and a data stream, wherein the code stream is a representation of the code portion of the virtual machine instruction, and wherein the data stream provides data that is needed to execute the code stream,

reading the code portion from the code stream;

directly accessing the data from the code stream; and

executing the virtual machine instruction using the data directly accessed form the code portion.

30. (Previously Presented) A method as recited in claim 29, wherein said converting the virtual machine instruction is performed at load time when a class file that includes the virtual machine instruction is loaded into the virtual machine.